

Large Neighborhood Local Search for the Maximum Set Packing Problem

Maxim Sviridenko* and Justin Ward†

Department of Computer Science, University of Warwick

Abstract

In this paper we consider the classical maximum set packing problem where set cardinality is upper bounded by k . We show how to design a variant of a polynomial-time local search algorithm with performance guarantee $(k+2)/3$. This local search algorithm is a special case of a more general procedure that allows to swap up to $\Theta(\log n)$ elements per iteration. We also design problem instances with locality gap $k/3$ even for a wide class of exponential time local search procedures, which can swap up to cn elements for a constant c . This shows that our analysis of this class of algorithms is almost tight.

1 Introduction

In this paper, we consider the problem of maximum unweighted k -set packing. In this problem, we are given a collection \mathcal{N} of n distinct k -element subsets of some ground set X . We say that two sets $A, B \in \mathcal{N}$ *conflict* if they share an element and call a collection of mutually non-conflicting sets from \mathcal{N} a *packing*. Then, the goal of the unweighted k -set packing problem is to find a packing $\mathcal{A} \subseteq \mathcal{N}$ of maximum cardinality. Here, we assume that each set has cardinality *exactly* k . This assumption is without loss of generality, since we can always add unique elements to each set of cardinality less than k to obtain such an instance.

The maximum set packing problem is one the basic optimization problems. It received a significant amount of attention from researchers in the last few decades (see e.g. [8]). It is known that a simple local search algorithm that starts with an arbitrary feasible solution and tries to add a constant number of sets to the current solution while removing a constant number of conflicting sets has performance guarantee arbitrarily close to $k/2$ [7]. It was also shown

*M.I.Sviridenko@warwick.ac.uk, Work supported by EPSRC grant EP/J021814/1, FP7 Marie Curie Career Integration Grant and Royal Society Wolfson Research Merit Award.

†J.D.Ward@dcs.warwick.ac.uk, Work supported by EPSRC grant EP/J021814/1.

in [7] that the analysis of such an algorithm is tight, i.e. there are maximum set covering instances where the ratio between a locally optimal solution value and the globally optimal solution value is arbitrarily close to $k/2$.

Surprisingly, Halldórsson[5] showed that if one increases the size of allowable swap to $\Theta(\log n)$ the performance guarantee can be shown to be at most $(k + 2)/3$. Recently, Cygan, Grandoni and Mastrolilli [3] improved the guarantee for the same algorithm to $(k + 1)/3$. This performance guarantee is the best currently known for the maximum set packing problem. The obvious drawback of these algorithms is that it runs in time $O(n^{\log n})$ and therefore its running time not polynomial.

Both algorithms rely only on the subset of swaps of size $\Theta(\log n)$ to be able to prove their respective performance guarantees. The Halldórsson's swaps are particularly well structured and have a straightforward interpretation in the graph theoretic language. In section 4 we employ techniques from fixed-parameter tractability to yield a procedure for finding well-structured improvements of size $O(\log n)$ in polynomial time. Our algorithm is based on color coding technique introduced by Alon, Yuster, and Zwick [1] and its extension by Fellows et al. [4], and solves a dynamic program to locate an improvement if one exists. Combining with Halldórsson's analysis, we obtain a polynomial time $\frac{k+2}{3}$ -approximation algorithm. In Section 6 we show that it is not possible to improve this result beyond $\frac{k}{3}$, even by choosing significantly larger improvements. Specifically, we construct a family of instances in which the locality gap for a local search algorithm applying all improvements of size t remains at least $\frac{k}{3}$ even when t is allowed to grow linearly with n . Our lower bound thus holds even for local search algorithms that are allowed to examine some exponential number of possible improvements at each stage.

2 A Quasi-Polynomial Time Local Search Algorithm

Let \mathcal{A} be a packing. We define an auxiliary multigraph $G_{\mathcal{A}}$ whose vertices correspond to sets in \mathcal{A} and whose edges correspond to sets in $\mathcal{N} \setminus \mathcal{A}$ that conflict with at most 2 sets in \mathcal{A} . That is, $E(G_{\mathcal{A}})$ contains a separate edge (S, T) for each set $X \in \mathcal{N} \setminus \mathcal{A}$ that conflicts with exactly two sets S and T in \mathcal{A} , and a loop on S for each set $X \in \mathcal{N}$ that conflicts with exactly one set S in \mathcal{A} . In order to simplify our analysis, we additionally say that each set $X \in \mathcal{A}$ conflicts with itself, and place such a loop on each set of \mathcal{A} . Note that $G_{\mathcal{A}}$ contains $O(n)$ vertices and $O(n)$ edges, for any value of \mathcal{A} .

Our local search algorithm uses $G_{\mathcal{A}}$ to search for improvements to the current solution \mathcal{A} . Formally, we call a set I of t edges in $G_{\mathcal{A}}$ a *t-improvement* if I covers at most $t - 1$ vertices of $G_{\mathcal{A}}$ and the sets of $\mathcal{N} \setminus \mathcal{A}$ corresponding to the edges in I are mutually disjoint.

Note that if I is a t -improvement for a packing \mathcal{A} , we can obtain a larger packing by removing the at most $t - 1$ sets covered by I from \mathcal{A} and then

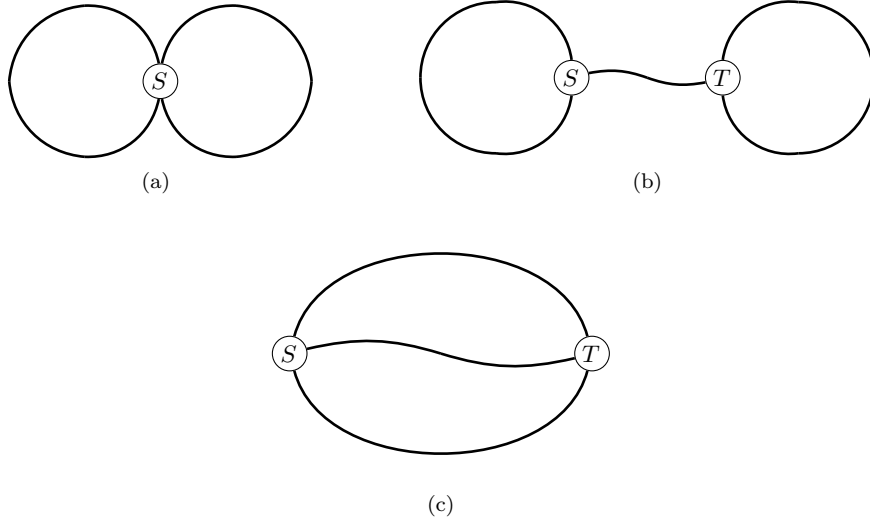


Figure 1: Canonical Improvements

adding the t sets corresponding to the edges of I to the result. We limit our search for improvements in $G_{\mathcal{A}}$ to those that exhibit the following particular form: an improvement is a *canonical* improvement if it forms a connected graph containing two distinct cycles. A general canonical improvement then comprises either 2 edge-disjoint cycles joined by a path, two edge-disjoint cycles that share a single vertex, or two distinct vertices joined by 3 edge-disjoint paths (see Figure 1).¹

Our algorithm, shown in Figure 2 proceeds by repeatedly calling the procedure $\text{IMPROVE}(G_{\mathcal{A}})$, which searches for a canonical $(4 \log n + 1)$ -improvement in the graph $G_{\mathcal{A}}$. Before searching for a canonical improvement, we first ensure that \mathcal{A} is a maximal packing by greedily adding sets from $\mathcal{N} \setminus \mathcal{A}$ to \mathcal{A} . If $\text{IMPROVE}(G_{\mathcal{A}})$ returns an improvement I , then I is applied to the current solution and the search continues. Otherwise, the current solution \mathcal{A} is returned.

In Section 3, we analyze the approximation performance of the local search algorithm under the assumption that $\text{IMPROVE}(G_{\mathcal{A}})$ always finds a canonical $(4 \log n + 1)$ -improvement, whenever such an improvement exists. In Section 4, we provide such an implementation $\text{IMPROVE}(G_{\mathcal{A}})$ that runs in deterministic polynomial time.

¹It can be shown that every t -improvement must contain a canonical t -improvement, and so we are not in fact restricting the search space at all by considering only canonical improvements. However, this fact will not be necessary for our analysis.

```

 $\mathcal{A} \leftarrow \emptyset$ 
loop
  for all  $S \in \mathcal{N} \setminus \mathcal{A}$  do
    if  $S$  does not conflict with any set of  $\mathcal{A}$  then
       $\mathcal{A} \leftarrow \mathcal{A} \cup \{S\}$ 
    end if
  end for

  Construct the auxiliary graph  $G_{\mathcal{A}}$  for  $\mathcal{A}$ 
   $I \leftarrow \text{IMPROVE}(G_{\mathcal{A}})$ 
  if  $I = \emptyset$  then
    return  $\mathcal{A}$ 
  else
     $\mathcal{A} \leftarrow (\mathcal{A} \setminus V(I)) \cup E(I)$ 
  end if
end loop

```

Figure 2: The General Local Search Procedure

3 Locality Gap of the Algorithm

In this section we prove the following upper bound on the locality gap for our algorithm. We consider an arbitrary instance \mathcal{N} of k -set packing, and let \mathcal{A} be the packing in \mathcal{N} produced by our local search algorithm and \mathcal{B} be any other packing in \mathcal{N} .

Theorem 3.1. $|\mathcal{B}| \leq \frac{k+2}{3}|\mathcal{A}|$.

For the purpose of our analysis, we consider the subgraph $H_{\mathcal{A},\mathcal{B}}$ of $G_{\mathcal{A}}$ consisting of only those edges of $G_{\mathcal{A}}$ corresponding to sets in \mathcal{B} . Then, every collection of edges in $H_{\mathcal{A},\mathcal{B}}$ is also present in $G_{\mathcal{A}}$. Moreover, because the edges of $H_{\mathcal{A},\mathcal{B}}$ all belong to the packing \mathcal{B} , any subset of them must be mutually disjoint. Thus, we can assume that no collection of at most $4 \log n + 1$ edges from $H_{\mathcal{A},\mathcal{B}}$ form any of the structures shown in Figure 1. Otherwise, the corresponding collection of edges in $G_{\mathcal{A}}$ would form a canonical $(4 \log n + 1)$ -improvement.

In order to prove Theorem 3.1, we make use of the following lemma of Berman and Fürer [2], which gives conditions under which the multigraph $H_{\mathcal{A},\mathcal{B}}$ must contain a canonical improvement.² We provide Berman and Fürer’s proof in the appendix.

Lemma 3.2 (Lemma 3.2 in [2]). *Assume that $|E| \geq \frac{p+1}{p}|V|$ in a multigraph $H = (V, E)$. Then, H contains a canonical improvement with at most $4p \log n - 1$ vertices.*

²Berman and Fürer call structures of the form shown in Figure 1 “binoculars.” Here, we have rephrased their lemma in our own terminology.

It will also be necessary to bound the total number of loops in $H_{\mathcal{A},\mathcal{B}}$. In order to do this, we shall consider a second auxiliary graph $H'_{\mathcal{A},\mathcal{B}}$ that is obtained from $H_{\mathcal{A},\mathcal{B}}$ in the following fashion:

Lemma 3.3. *Let $H = (V, E)$ be a multigraph and let $H' = (V', E')$ be obtained from H by deleting all vertices of H with loops on them and, for each edge with one endpoint incident to a deleted vertex, introducing a new loop on this edge's remaining vertex. Let $t \geq 3$. Then, if H' contains a canonical t -improvement, H contains a canonical $(t + 2)$ -improvement.*

A proof of Lemma 3.3, based on a sketch given by Halldórsson [5], appears in the appendix.

We now turn to the proof of Theorem 3.1. Every set in \mathcal{B} must conflict with some set in \mathcal{A} , or else \mathcal{A} would not be maximal. We partition the sets of \mathcal{B} into three collections of sets, depending on how many sets in \mathcal{A} they conflict with. Let $\mathcal{B}_1, \mathcal{B}_2$, and \mathcal{B}_3 be collections of those sets of \mathcal{B} that conflict with, respectively, exactly 1, exactly 2, and 3 or more sets in \mathcal{A} (note that each set of $\mathcal{A} \cap \mathcal{B}$ is counted in \mathcal{B}_1 , since we have adopted the convention that such sets conflict with themselves).

Because each set in \mathcal{A} contains at most k elements and the sets in \mathcal{B} are mutually disjoint, we have the inequality

$$|\mathcal{B}_1| + 2|\mathcal{B}_2| + 3|\mathcal{B}_3| \leq k|\mathcal{A}|. \quad (1)$$

We now bound the size of \mathcal{B}_1 and \mathcal{B}_2 .

Let \mathcal{A}_1 be the collection of sets from \mathcal{A} that conflict with sets of \mathcal{B}_1 . Then, note that each set of \mathcal{B}_1 corresponds to a loop in $H_{\mathcal{A},\mathcal{B}}$ and the sets of \mathcal{A}_1 correspond to the vertices on which these loops occur. Any vertex of $H_{\mathcal{A},\mathcal{B}}$ with two loops would form an improvement of the form shown in Figure 1a. Thus, each vertex in $H_{\mathcal{A},\mathcal{B}}$ has at most 1 loop and hence:

$$|\mathcal{B}_1| = |\mathcal{A}_1|. \quad (2)$$

Now, we show that $|\mathcal{B}_2| \leq 2|\mathcal{A} \setminus \mathcal{A}_1|$. By way of contradiction, suppose that $|\mathcal{B}_2| \geq 2|\mathcal{A} \setminus \mathcal{A}_1|$. We construct an auxiliary graph $H'_{\mathcal{A},\mathcal{B}}$ from $H_{\mathcal{A},\mathcal{B}}$ as in Lemma 3.3. The number of edges in this graph is exactly $|\mathcal{B}_2|$ and the number of vertices is exactly $|\mathcal{A} \setminus \mathcal{A}_1|$. Thus, if $|\mathcal{B}_2| \leq 2|\mathcal{A} \setminus \mathcal{A}_1|$, then from Lemma 3.2 (with $p = 1$), there is a canonical improvement in $H'_{\mathcal{A},\mathcal{B}}$ of size at most $4 \log n - 1$. But, from Lemma 3.3 this means there must be a canonical improvement in $H_{\mathcal{A},\mathcal{B}}$ of size at most $4 \log n + 1$, contradicting the local optimality of \mathcal{A} . Thus,

$$|\mathcal{B}_2| < 2|\mathcal{A} \setminus \mathcal{A}_1| \quad (3)$$

Adding (1), twice (2), and (3), we obtain

$$3|\mathcal{B}_1| + 3|\mathcal{B}_2| + 3|\mathcal{B}_3| \leq k|\mathcal{A}| + 2|\mathcal{A}_1| + 2|\mathcal{A} \setminus \mathcal{A}_1|,$$

which implies that $3|\mathcal{B}| \leq (k + 2)|\mathcal{A}|$.

4 Finding Canonical Improvements

A naïve implementation of the local search algorithm described in Section 2 would run in only quasi-polynomial time, since at each step there are $n^{\Omega(\log n)}$ possible improvements of size $t = 4 \log n + 1$. In contrast, we now show that it is possible to find a *canonical* improvement of size t in polynomial time whenever one exists.

We first give a randomized algorithm, using the color coding approach of Alon, Yuster, and Zwick [1]. If some t -improvement exists, our algorithm finds it with polynomially small probability. In Section 5, we show how to use this algorithm to implement a local search algorithm that succeeds with high probability, and how to obtain a deterministic variant via derandomization.

We now describe the basic, randomized color coding algorithm. Again, consider an arbitrary instance \mathcal{N} of k -set packing and let X be the ground set of \mathcal{N} . Let K be a collection of kt colors. We assign each element of X a color from K uniformly at random, and assign each k -set from \mathcal{N} the set of all its elements' colors. We say that a collection of sets $\mathcal{A} \subseteq \mathcal{N}$ is *colorful* if no color appears twice amongst the sets of \mathcal{A} . We note that if a collection of sets \mathcal{A} is colorful, then \mathcal{A} must form a packing, since no two sets in \mathcal{A} can share an element.

We assign each edge of $G_{\mathcal{A}}$ the same set of colors as its corresponding set in \mathcal{N} , and, similarly, say that a collection of edges is colorful if the corresponding collection of sets from \mathcal{N} is colorful. Now, we consider a subgraph of $G_{\mathcal{A}}$ made up of some set of at most t edges I . If this graph has the one of the forms shown in Figure 1 and I is colorful, then I must be a canonical t -improvement. We now show that, although the converse does not hold, our random coloring makes any given canonical t -improvement colorful with probability only polynomially small in n .

Consider a canonical improvement I of size $1 \leq i \leq t$. The i sets corresponding to the edges of I must be disjoint and so consist of ki separate elements. The probability that I is colorful is precisely the probability that all of these ki elements are assigned distinct colors. This probability can be estimated as

$$\frac{\binom{kt}{ki} (ki)!}{(kt)^{ki}} = \frac{(kt)!}{(kt - ki)! (kt)^{ki}} \geq \frac{(kt)!}{(kt)^{kt}} > e^{-kt} = e^{-4k \log n - k} = e^{-k} n^{-8k}, \quad (4)$$

where in the last line, we have used the fact that $e^{\log n} = e^{\ln n \log e} = n^{\log e} < n^2$.

We now show how to use this random coloring to find canonical improvements in $G_{\mathcal{A}}$. Our approach is based on finding colorful paths and cycles and employs dynamic programming.

We give a dynamic program that, given a coloring for edges of $G_{\mathcal{A}}$, as described above, finds a colorful path of length at most t in $G_{\mathcal{A}}$ between each pair of vertices S and T , if such a path exists. For each vertex S and T of $G_{\mathcal{A}}$, each value $i \leq t$, and each set C of ki colors from K , we have an entry $\mathcal{D}(S, T, i, C)$ that records whether or not there is some colorful path of length i between S and T whose edges are colored with precisely those colors in C . In our table, we explicitly include the case that $S = T$.

We compute the entries of \mathcal{D} bottom-up in the following fashion. We set $\mathcal{D}(S, T, 0, C) = 0$ for all pairs of vertices S, T , and C . Then, we compute the entries $\mathcal{D}(S, T, i, C)$ for $i > 0$ as follows. We set $\mathcal{D}(S, T, i, C) = 1$, if there is some edge (V, T) incident to vertex T in $G_{\mathcal{A}}$ such that (V, T) is colored with a set of k distinct colors $B \subseteq C$ and the entry $\mathcal{D}(S, V, i-1, C \setminus B) = 1$. Otherwise, we set $\mathcal{D}(S, T, i, C) = 0$.

To determine if $G_{\mathcal{A}}$ contains a colorful path of given length $i \leq t$ from S to T , we simply check whether $\mathcal{D}(S, T, i, C) = 1$ for some set of colors C . Similarly, we can use our dynamic program to find colorful *cycles* of length j that include some given vertex U by consulting $\mathcal{D}(U, U, j, C)$ for each set of colors C . The actual path or cycle can then be found by backtracking through the table \mathcal{D} . We note that while the cycles and paths found by this procedure are not necessarily simple, they are edge-disjoint.

For each value of i , there are at most $n^2 \binom{kt}{ki}$ entries in $\mathcal{D}(S, T, i, C)$. To compute each such entry, we examine each edge (V, T) incident to T , check if (V, T) is colored with a set of k colors $B \subseteq C$ and consult $\mathcal{D}(S, T, i, C \setminus B)$, all which can be accomplished in time $O(nki)$. Thus, the total time to compute \mathcal{D} up to $i = t$ is of order:

$$\sum_{i=1}^t n^3 ki \binom{kt}{ki} \leq n^4 kt 2^{kt}$$

In order to find a canonical t -improvement, we first compute the table \mathcal{D} up to $i = t$. Then, we search for improvements of each kind shown in Figure 1 by enumerating over all choices of S and T , and looking for an appropriate collection of cycles or paths involving these vertices that use mutually disjoint sets of colors. Specifically:

- To find improvements of the form shown in Figure 1a, we enumerate over all n vertices S . For all disjoint sets of ka and kb colors C_a and C_b with $a + b \leq t$, we check if $\mathcal{D}(S, S, a, C_a) = 1$ and $\mathcal{D}(S, S, b, C_b) = 1$. This can be accomplished in time

$$n \sum_{a=1}^t \sum_{b=1}^{t-a} 2^{ka} 2^{kb} kt = O(nkt^3 2^{kt})$$

- To find improvements of the form shown in Figure 1b we enumerate over all distinct vertices S and T . For all disjoint sets of ka , kb , and kc colors C_a, C_b , and C_c with $|C_a| + |C_b| + |C_c| \leq t$, we check if $\mathcal{D}(S, S, a, C_a) = 1$, $\mathcal{D}(T, T, b, C_b) = 1$, and $\mathcal{D}(S, T, c, C_c) = 1$. This can be accomplished in time

$$n^2 \sum_{a=1}^t \sum_{b=1}^{t-a} \sum_{c=1}^{t-a-b} 2^{ka} 2^{kb} 2^{kc} kt = O(n^2 kt^4 2^{kt})$$

- To find improvements of the form shown in Figure 1c we again enumerate over all distinct vertices S and T . For all disjoint sets of ka , kb , and kc colors C_a, C_b , and C_c with $|C_a| + |C_b| + |C_c| \leq t$, we check if $\mathcal{D}(S, T, a, C_a) =$

1, $\mathcal{D}(S, T, b, C_b) = 1$, and $\mathcal{D}(S, T, c, C_c) = 1$. This can be accomplished in time

$$n^2 \sum_{a=1}^t \sum_{b=1}^{t-a} \sum_{c=1}^{t-a-b} 2^{ka} 2^{kb} 2^{kc} kt = O(n^2 kt^4 2^{kt})$$

Thus, the total time spent searching for a canonical t -improvement is then at most:

$$O(n^2 kt 2^{kt} + nkt^3 2^{kt} + 2n^2 kt^4 2^{kt}) = O(n^2 kt^4 2^{kt}) = O(2^k k \cdot n^{4k+2} \log^4 n).$$

5 The Deterministic, Large Neighborhood Local Search Algorithm

The analysis of the local search algorithm in Section 3 supposed that every call to $\text{IMPROVE}(G_{\mathcal{A}})$ returns \emptyset only when no canonical t -improvement exists in $G_{\mathcal{A}}$. Under this assumption, the algorithm is a $\frac{k+2}{3}$ -approximation. In contrast, the dynamic programming implementation given in Section 4 may fail to find a canonical improvement I if the chosen random coloring does not make I colorful. As we have shown in (4), this can happen with probability at most $(1 - e^{-k} n^{-8k})$.

Suppose that we implement each call to $\text{IMPROVE}(G_{\mathcal{A}})$ by running the algorithm of Section 4 $cN = ce^k n^{8k} \ln n$ times, each with a different random coloring. We now show that the resulting algorithm is a polynomial time $\frac{k+2}{3}$ -approximation with high probability $1 - n^{1-c}$.

We note that each improvement found by our local search algorithm must increase the size of the packing \mathcal{A} , and so the algorithm makes at most n calls to $\text{IMPROVE}(G_{\mathcal{A}})$. We set $N = e^k n^{8k+1} \ln n$, and then implement each such call by repeating the color coding algorithm of Section 4 cN times for some $c > 1$, each with a new random coloring. The probability that any given call $\text{IMPROVE}(G_{\mathcal{A}})$ succeeds in finding a canonical t -improvement when one exists is then at least:

$$1 - (1 - e^{-k} n^{-8k})^{cN} \geq 1 - \exp\{e^{-k} n^{-8k} \cdot ce^k n^{8k} \ln n\} = 1 - n^{-c}.$$

And so, the probability that all calls to $\text{IMPROVE}(G_{\mathcal{A}})$ satisfy the assumptions of Theorem 3.1 is at least:

$$(1 - n^{-c})^n \geq 1 - n^{1-c}$$

The resulting algorithm is therefore a $\frac{k+2}{3}$ -approximation with high probability. It requires at most n calls to $\text{IMPROVE}(G_{\mathcal{A}})$, each requiring total time

$$O(cN \cdot 2^k k n^{4k+2} \log^4 n) = O(c(2e)^k k n^{12k+2} \log^n n \ln n) = cn^{O(k)}$$

Using the general approach described by Alon, Yuster, and Zwick [1], we can in fact give a *deterministic* implementation of $\text{IMPROVE}(G_{\mathcal{A}})$, which *always* succeeds in finding a canonical t -improvement in $G_{\mathcal{A}}$ if such an improvement

exists. Rather than choosing a coloring of the ground set X at random, we use a collection \mathcal{K} of colorings (each of which is given as a mapping $X \rightarrow K$) with the property that every canonical t -improvement $G_{\mathcal{A}}$ is colorful with respect to some coloring in \mathcal{K} . For this, it is sufficient to find a collection of \mathcal{K} of colorings such that for every set of at most kt elements in X , there is some coloring in \mathcal{K} that assigns these kt elements kt distinct colors from K . Then, we implement $\text{IMPROVE}(G_{\mathcal{A}})$ by running the dynamic programming algorithm of Section 5 on each such coloring, and returning the first improvement found. Because every canonical t -improvement contains at most kt distinct elements of the ground set, every such improvement must be made colorful by some coloring in \mathcal{K} , and so $\text{IMPROVE}(G_{\mathcal{A}})$ will always find a canonical t -improvement if one exists.

We now show how to construct the desired collection of colorings \mathcal{K} by using a *kt-perfect family* of hash functions from $X \rightarrow K$. Briefly, a perfect hash function for a set $S \subseteq A$ is a mapping from A to B that is one-to-one on S . A p -perfect family is then collection of perfect hash functions, one for each set $S \subseteq A$ of size at most p . Building on work by Fredman, Komlós and Szemerédi [10] and Schmidt and Siegal [9], Alon and Naor show (in Theorem 3 of [11]) how to explicitly construct a perfect hash function from $[m]$ to $[p]$ for some $S \subset [m]$ of size p in time $\tilde{O}(p \log m)$. This hash function is described in $O(p + \log p \cdot \log \log m)$ bits. The maximum size of a p -perfect family of such functions is therefore $2^{O(p + \log p \cdot \log \log m)}$. Moreover, the function can be evaluated in time $O(\log m / \log p)$.

Then, we can obtain a deterministic, polynomial time $\frac{k+2}{3}$ approximation as follows. Upon receiving the set packing instance \mathcal{N} with ground set X , we compute a kt -perfect family \mathcal{K} of hash functions from X to a set of kt colors K . Then, we implement each call to $\text{IMPROVE}(G_{\mathcal{A}})$ as described, by enumerating over the colorings in \mathcal{K} . We note that since each set in \mathcal{N} has size k , $|X| \leq |\mathcal{N}|k = nk$, so each improvement makes at most

$$2^{O(kt + \log kt \cdot \log \log kn)} = 2^{O(k \log n + \log(k \log n) \cdot \log \log kn)} = n^{O(k)}$$

calls to the dynamic programming algorithm of Section 5 (one per coloring in \mathcal{K}) and each of these calls takes time at most $n^{O(k)}$ (including the time to evaluate the coloring on each element of the ground set). Moreover, the initial construction of \mathcal{K} takes time at most $2^{kt} \tilde{O}(kt \log kn) = n^{O(k)}$.

6 A Lower Bound

We now show that our analysis is almost tight. Specifically, we show that the locality gap of t -local search is least $\frac{k}{3}$, even when t is allowed to grow on the order of n .

Theorem 6.1. *Let $c = \frac{9}{2e^{5k}}$ and suppose that $t \leq cn$ for all sufficiently large n . There, there exist 2 pairwise disjoint collections of k -sets \mathcal{S} and \mathcal{O} with $|\mathcal{S}| = 3n$ and $|\mathcal{O}| = kn$ such that any collection of $a \leq t$ sets in \mathcal{O} conflict with at least a sets in \mathcal{S} .*

In order to prove Theorem 6.1 we make use of the following (random) construction. Let X be a ground set of $3kn$ elements, and consider a collection \mathcal{S} of $3n$ sets, each containing k distinct elements of X . We construct a random collection \mathcal{R} of kn disjoint subsets of X , each containing 3 elements.

The number of distinct collections generated by this procedure is equal to the number of ways to partition the $3kn$ elements of X into kn disjoint 3-sets. We define the quantity $\tau(m)$ to be the number of ways that m elements can be divided into $m/3$ disjoint 3-sets:

$$\tau(m) \triangleq \frac{m!}{(3!)^{m/3}(m/3)!}.$$

In order to verify the above formula, consider the following procedure for generating a random partition. We first arrange the m elements in some order and then make a 3-set from the elements at positions $3i$, $3i - 1$ and $3i - 2$, for each $i \in [\frac{m}{3}]$ (that is, we group each set of 3 consecutive elements in the ordering into a triple). Now, we note that two permutations of the elements produce the same partition if they differ only in the ordering of the 3 elements within each of the $m/3$ triples or in the ordering of the $m/3$ triples themselves. Thus, each partition occurs in exactly $(3!)^{m/3}(m/3)!$ of the $m!$ possible orderings.

The probability that any particular collection of a disjoint 3-sets occurs in \mathcal{R} is given by

$$p(a) \triangleq \frac{\tau(3kn - 3a)}{\tau(3kn)}.$$

This is simply the number of ways to partition the remaining $3kn - 3a$ elements into $kn - a$ disjoint 3-sets, divided by the total number of possible partitions of all $3kn$ elements.

We say that a collection \mathcal{A} of a sets in \mathcal{S} is *unstable* if there is some collection \mathcal{B} of at least a sets in \mathcal{R} that conflict with only those sets in \mathcal{A} . Note that there is an improvement of size a for \mathcal{S} only if there is some unstable collection \mathcal{A} of size a in \mathcal{S} .³ We now derive an upper bound on the probability that our random construction of \mathcal{R} results in a given collection \mathcal{A} in \mathcal{S} being unstable.

Lemma 6.2. *A collection of a sets in \mathcal{S} is unstable with probability less than $\frac{\binom{ka}{3da}\binom{kn}{da}}{\binom{3kn}{3da}}$.*

Proof. A collection \mathcal{A} of a k -sets from \mathcal{S} is unstable precisely when there is a collection \mathcal{B} of a 3-sets in \mathcal{R} that contain only those $k(a - 1)$ elements appearing in the sets of \mathcal{A} . There are $\binom{ka}{3a}$ ways to choose the $3a$ elements from which we construct \mathcal{B} . For each such choice, there are $\tau(3a)$ possible ways to partition the elements into 3-sets, each occurring with probability $p(3a) = \tau(3kn - 3a)/\tau(3n)$.

³In fact, for an improvement to exist, there must be some collection \mathcal{B} of $a + 1$ such sets in \mathcal{R} . This stronger condition is unnecessary for our bound, however.

Applying the union bound, the probability that \mathcal{A} is unstable is then at most:

$$\begin{aligned}
\binom{ka}{3a} \tau(3a) \frac{\tau(3kn-3a)}{\tau(3kn)} &= \binom{ka}{3a} \frac{(3a)!}{(3!)^a a!} \cdot \frac{(3(kn-a))!}{(3!)^{kn-a} (kn-a)!} \cdot \frac{(3!)^{kn} (kn)!}{(3kn)!} \\
&= \binom{ka}{3a} \frac{(3a)!(3(kn-a))!}{(3kn)!} \frac{(kn)!}{(kn-a)! a!} \\
&= \frac{\binom{ka}{3a} \binom{kn}{a}}{\binom{3kn}{3a}}
\end{aligned}$$

□

Theorem 6.1. Let U_a be number of unstable collections of size a in \mathcal{S} , and consider $\mathbb{E}[U_a]$. There are precisely $\binom{3n}{a}$ such collections, and from Lemma 6.2, each occurs with probability less than $\frac{\binom{ka}{3a} \binom{kn}{a}}{\binom{3kn}{3a}}$. Thus:

$$\mathbb{E}[U_a] < \frac{\binom{3n}{a} \binom{ka}{3a} \binom{kn}{a}}{\binom{3kn}{3a}} \quad (5)$$

Applying the upper and lower bounds

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k,$$

in the numerator and denominator, respectively, of (5), we obtain the upper bound

$$\frac{(e3n)^a}{a^a} \cdot \frac{(eka)^{3a}}{(3a)^{3a}} \cdot \frac{(ekn)^a}{a^a} \cdot \frac{(3a)^{3a}}{(3kn)^{3a}} = \left(\frac{e^5 3^4 k^4 a^6 n^2}{3^6 k^3 a^5 n^3}\right)^a = \left(\frac{e^5 ka}{9n}\right)^a.$$

Then, the expected number of unstable collections in \mathcal{S} of size *at most* t (and hence the expected number of t -improvements for \mathcal{S}) is less than

$$\sum_{a=1}^t \mathbb{E}[U_a] < \sum_{a=1}^t \left(\frac{e^5 ka}{9n}\right)^a. \quad (6)$$

For all sufficiently large n , we have $a \leq t \leq cn$ and so

$$\sum_{a=1}^t \left(\frac{e^5 ka}{9n}\right)^a \leq \sum_{a=1}^t \left(\frac{e^5 kcn}{9n}\right)^a = \sum_{a=1}^t \left(\frac{e^5 k}{9} \frac{9}{2e^5 k}\right)^a = \sum_{a=1}^t \left(\frac{1}{2}\right)^a < 1.$$

Thus, there must exist some collection \mathcal{O} in the support of \mathcal{R} that creates no unstable collections of size at most t in \mathcal{S} . Then, \mathcal{O} is a collection of pairwise disjoint sets of size kn satisfying the conditions of the theorem. □

7 Conclusion

We have given a polynomial time $\frac{k+2}{3}$ approximation algorithm for the problem of k -set packing. Our algorithm is based on a simple local search algorithm, but incorporates ideas from fixed parameter tractability to search large neighborhoods efficiently, allowing us to achieve an approximation guarantee exceeding the $k/2$ bound of Hurkens and Schrijver [7]. In contrast, our lower bound of $k/3$ shows that local search algorithms considering still larger neighborhoods, including neighborhoods of exponential size, can yield only slight improvements.

An interesting direction for future research would be to close the gap between our $k/3$ lower bound and $\frac{k+2}{3}$ upper bound. Recently, Cygan, Grandoni, and Mastrolilli [3] have given a quasi-polynomial time local search algorithm attaining an approximation ratio of $(k+1)/3$. Their analysis is also based on that of Berman and Fürer [2] and Halldórsson [5], but their algorithm requires searching for improvements with a more general structure than those that we consider, and it is unclear how to apply similar techniques as ours in this case. Nevertheless, we conjecture that it is possible to attain an approximation ratio of $\frac{k+1}{3}$ in polynomial time, although this will likely require more sophisticated techniques than we consider here.

In contrast to all known positive results, the best known NP-hardness result for k -set packing is, due to Hazan, Safra, and Schwartz [6], is only $O(k/\ln k)$. A more general open problem is whether the gap between this result and algorithmic results can be narrowed.

Finally, we ask whether our results can be generalized to the independent set problem in $(k+1)$ -claw free graphs. Most known algorithms for k -set packing, including those given by Halldórsson [5] and Cygan, Grandoni, and Mastrolilli [3] generalized trivially to this setting. However, this does not seem to be the case for the color coding approach that we employ, as it relies on the set packing representation of problem instances.

Acknowledgements

We would like to thank Oleg Pikhurko for extremely enlightening discussion on random graphs.

References

- [1] N. Alon, R. Yuster, and U. Zwick, Color-coding, *Journal of the ACM*, 42(4), pp. 844–856 (1995)
- [2] P. Berman and M. Fürer, Approximating maximum independent set in bounded degree graphs, *SODA* (1994)
- [3] M. Cygan, F. Grandoni, and M. Mastrolilli, How to sell hyperedges: the hypermatching assignment problem, *SODA* (2013)
- [4] M. Fellows, C. Knauer, and N. Nishimura, Faster fixed-parameter tractable algorithms for matching and packing problems, *Algorithmica*, pp. 167–176 (2004)

- [5] M. Halldórsson, Approximating Discrete Collections via Local Improvements, SODA, (1995)
- [6] Hazan, S. Safra, and O. Schwartz, On the complexity of approximating k-set packing. Computational Complexity, 15(1), pp. 2039 (2006)
- [7] C. Hurkens and A. Schrijver, On the Size of Systems of Sets Every t of Which Have an SDR, with an Application to the Worst-Case Ratio of Heuristics for Packing Problems, SIAM Journal of Discrete Mathematics 2(1), pp. 68–72 (1989)
- [8] V. Paschos, A survey of approximately optimal solutions to some covering and packing problems, ACM Computing Surveys 29(2), pp. 171 – 209 (1997)
- [9] J.P. Schmidt and A. Siegel, The spatial complexity of oblivious k-probe hash functions. SIAM Journal on Computing, 19(5), pp. 775–786 (1990)
- [10] M. Fredman, J. Komlós, and E. Szemerédi, Storing a sparse table with $O(1)$ worst case access time. Journal of the ACM, 31(3), pp. 538–544, (1984)
- [11] N. Alon and M. Naor, Derandomization, witnesses for boolean matrix multiplication and construction of perfect hash functions. Algorithmica, 16(4/5), pp. 434–449 (1996)

A Appendix

Here we provide detailed proofs of the cited technical results.

Lemma 3.2 (Lemma 3.2 in [2]). *Assume that $|E| \geq \frac{p+1}{p}|V|$ in a multigraph $H = (V, E)$. Then, H contains a canonical improvement with at most $4p \log n - 1$ vertices.*

Proof. Suppose that H' is the smallest induced subgraph of H that satisfies the condition

$$E(H') \geq \frac{p}{p+1}V(H'). \quad (7)$$

We shall show that H' must contain a canonical improvement. First, we note that H' cannot contain any degree 1 vertices. Otherwise, we could remove all such vertices to obtain a smaller graph satisfying (7). Moreover, any chain of degree 2 vertices in H' has fewer than p vertices. Otherwise, we could remove this chain of p vertices, together with the $p+1$ edges incident on them to obtain a smaller graph satisfying (7). We replace every chain of degree 2 vertices in H' with a single edge connecting its the endpoints to obtain a graph H_3 with minimum degree 3.

Let I_3 be a minimal connected subgraph of H_3 with exactly 2 distinct cycles. Then, I_3 is a canonical improvement in H_3 and $|V(I_3)| \leq |E(I_3)| + 1$. By expanding each contracted chains of vertices in I_3 , we obtain a connected subgraph of H' that contains 2 distinct cycles. Then, $|V(I')| \leq p(|V(I_3)| + 1)$. Thus, to complete the proof of Lemma 3.2 it suffices to show that H_3 must contain a connected subgraph I_3 with exactly 2 distinct cycles and $|V(I_3)| \leq 4 \log n - 1$.

Let $n = |V|$ and note that $|V(H_3)| \leq |V(H')| \leq V(H) \leq n$. We first note that H_3 has maximum girth less than $2 \log n$. To prove this, simply construct

a breadth first search tree rooted at some vertex in the graph. There must be some vertex v of distance less than $\log n$ from the root without 2 children. Since v has degree 3, it must have an edge to some previously visited vertex u in the tree (where possibly $u = v$). The paths from u and v to the root, together with the edge (u, v) form a connected subgraph C that has at most $2 \log n$ vertices and contains both the root of the tree and a cycle. Let C be a minimal such subgraph. We contract C to a single vertex and call the resulting graph H'_3 . Then, H'_3 must also minimum degree 3 and at most n vertices. Repeating the argument we can find a minimal subgraph C' in H'_3 with at most $2 \log n$ vertices that contains both the root of H'_3 and a cycle. Let I_3 be the induced subgraph of H_3 containing the vertices of C and C' . Then, I_3 has at most $4 \log n - 1$ vertices, and is a connected subgraph of H_3 containing exactly 2 distinct cycles. \square

Lemma 3.3. *Let $H = (V, E)$ be a multigraph and let $H' = (V', E')$ be obtained from H by deleting all vertices of H with loops on them and, for each edge with one endpoint incident to a deleted vertex, introducing a new loop on this edge's remaining vertex. Let $t \geq 3$. Then, if H' contains a canonical t -improvement, H contains a canonical $(t + 2)$ -improvement.*

Proof. Our argument is based on a sketch given by Halldórsson [5]. In the interest of completeness, we present a more detailed argument here.

Note that any canonical t improvement in H' that does not contain a loop is also present in H . Consider, then, a canonical t -improvement I in H' which contains either one or two loops (i.e. an improvement of the form 1a and 1b, where one or both of the cycles are loops). A loop on vertex v in H' corresponds to an edge (u, v) in H , where v has a loop. The improvement I in H' must have 2 cycles joined by either a path or a single vertex. Figure 3 illustrates all of the possible configurations for I in H' and the related canonical improvements in H , which we now show must exist.

If exactly one of these cycles is a loop on some vertex v , then we must have a path (possibly of length 0) joining v to another cycle J in H' . This path and J are also present in H . Additionally, in H we must have an edge (u, v) and a loop on u . Thus, the loop on u , together with the edge u, v , the cycle J and the path connecting v to J form a canonical improvement with only one more edge than I .

Now, suppose that both of these cycles are loops on some vertices v_1 and v_2 (where possibly $v_1 = v_2$). If the corresponding edges (v_1, u_1) and (v_2, u_2) in H have distinct endpoints $u_1 \neq u_2$, then the two loops on u_1 and u_2 together with these edges and the path (of length 0, in the case that $v_1 = v_2$) joining v_1 and v_2 form a canonical improvement. If $u_1 = u_2$, then the edges (v_1, u_1) and (v_2, u_2) , together with the path (again, of length 0 if $v_1 = v_2$) from v_1 to v_2 forms a cycle. This, together with the loop on the vertex $u_1 = u_2$, forms a canonical improvement. In both cases, the canonical improvement in H has only 2 more edges than I . \square

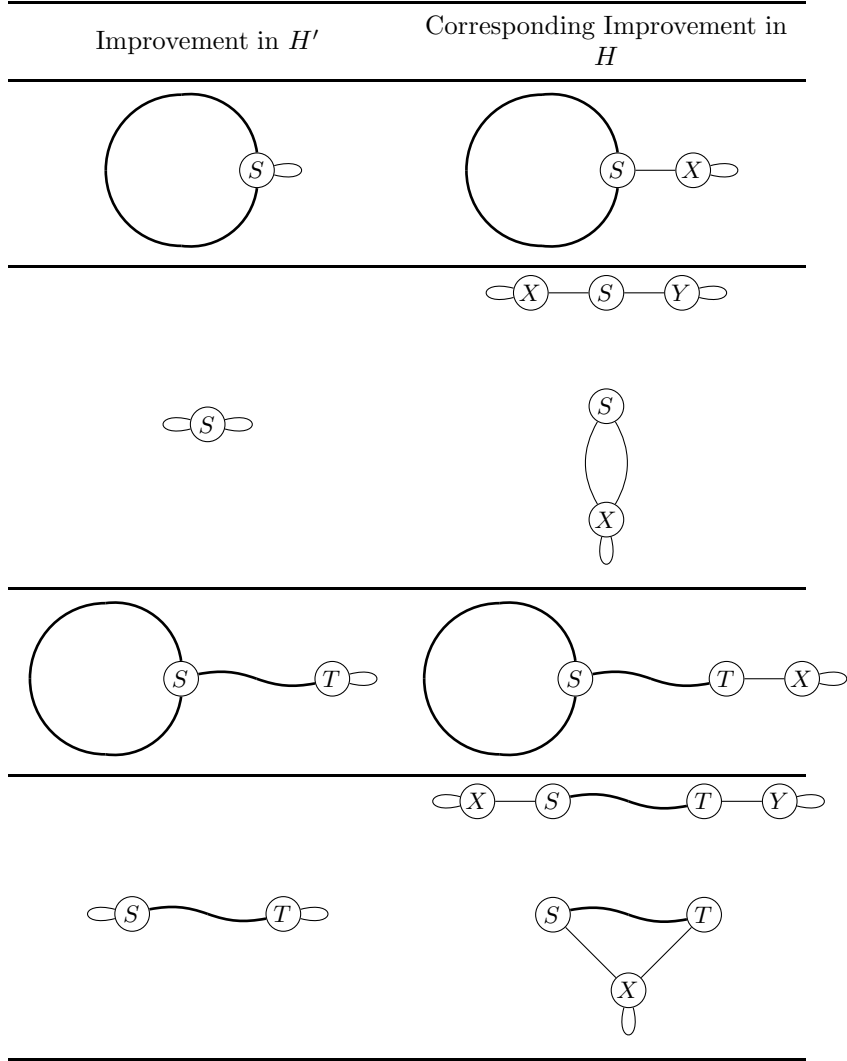


Figure 3: Canonical improvements in H' and corresponding improvements in H .